

Working with Date and Time

1	Get system values	2
1.1	Now Function	2
1.2	Date Function	2
1.3	Time Function.....	2
2	Get values from Date.....	4
2.1	Extract values	4
2.1.1	Day Function	4
2.1.2	Month Function	4
2.1.3	Year Function	5
2.1.4	Weekday Function	5
2.1.5	WeekdayName Function	5
2.1.6	MonthName Function	6
2.2	Return a date	6
2.2.1	DateSerial Function.....	6
2.2.2	DateValue Function.....	7
2.3	Set values.....	7
2.3.1	Date Statement	7
3	Get values from Time	7
3.1	Extract values	7
3.1.1	Hour Function.....	7
3.1.2	Minute Function.....	7
3.1.3	Second Function	8
3.2	Return a time	8
3.2.1	TimeSerial Function	8
3.2.2	TimeValue Function	8
3.3	Set values.....	9
3.3.1	Time Statement.....	9
4	Date and Time calculations	9
4.1	DateAdd Function.....	9
4.2	DateDiff Function.....	10
4.3	DatePart Function	11
5	Formating	12
5.1	Format Function	12
5.2	FormatDateTime Function.....	15

1 Get system values

1.1 Now Function

Now

Returns a Variant (Date) specifying the current date and time according your computer's system date and time.

1.2 Date Function

Date

Returns a Variant (Date) containing the current system date.

1.3 Time Function

Time

Returns a Variant (Date) indicating the current system time.

Example:

[CODE]

```
' =====  
Private Sub Form_Load()  
    Me.AutoRedraw = True  
    Print "Now ==>"; Tab(20); Now  
    Print "Date ==>"; Tab(20); Date  
    Print "Time ==>"; Tab(20); Time  
End Sub  
' =====  
[/CODE]
```

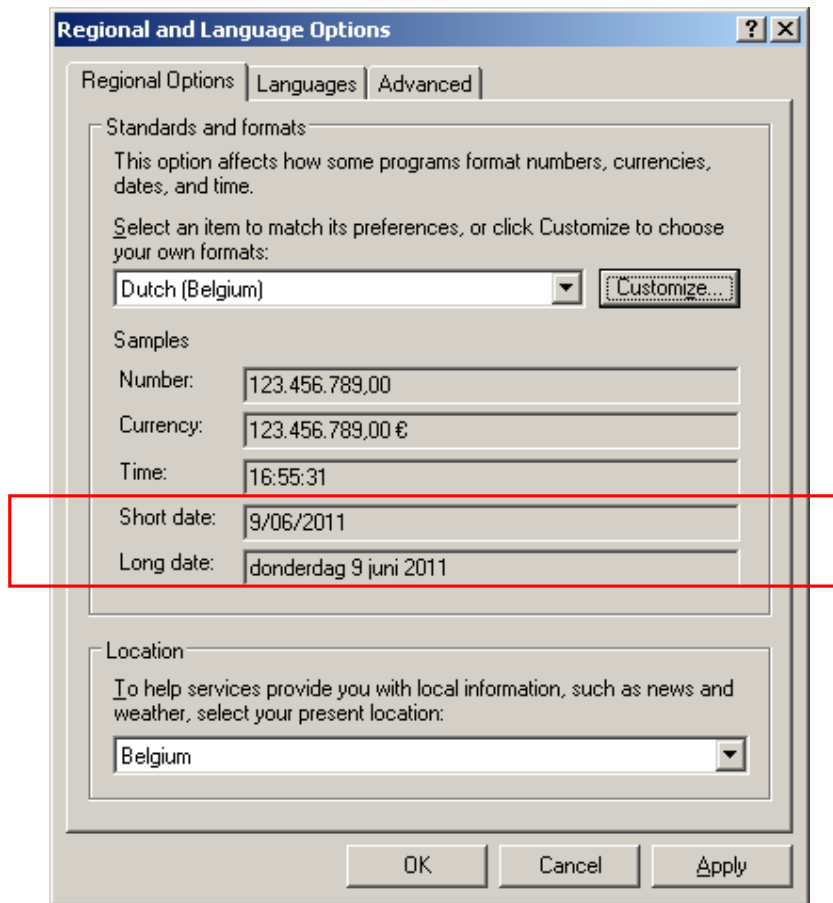
Remark:

The format of the data is according to the settings in the system.

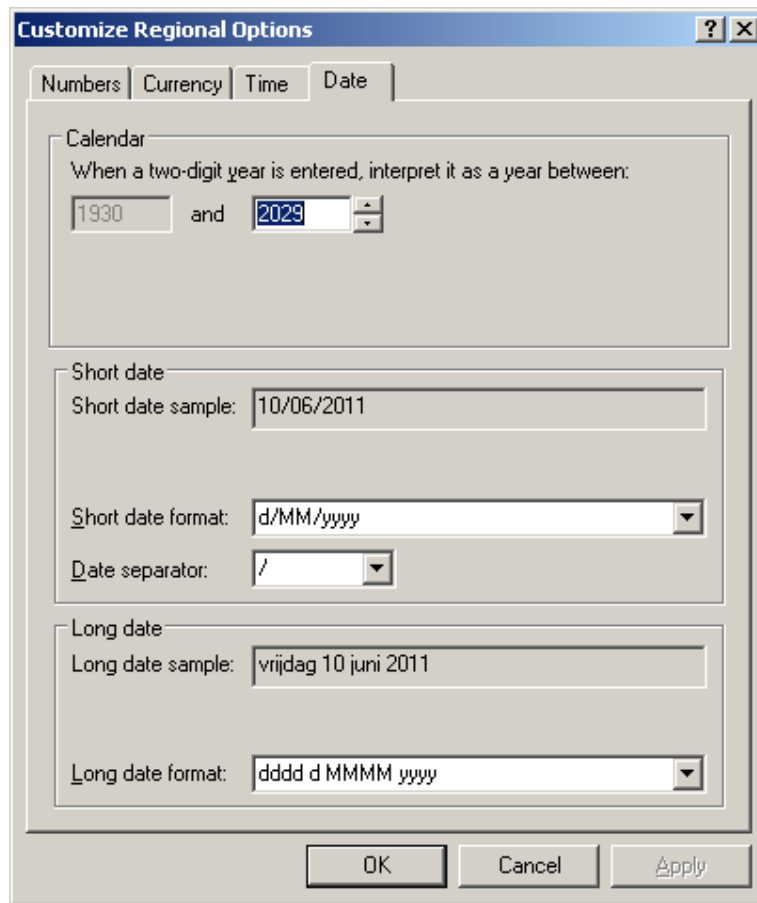
For XP in:

Click on 'Start' – 'Control panel' – 'Regional and Language Options'.

See frame 'Standards and Formats'



For 'Now' and 'Date': the date is in the format of 'Short date'
It's possible to modify this by clicking on the button [Customize...] and selecting the tab: 'Date'
It's possible to experiment with the different formats (see later)



2 Get values from Date

2.1 Extract values

2.1.1 Day Function

Day (date)

Returns a Variant (Integer) specifying a whole number between 1 and 31, inclusive, representing the day of the month.

Arguments:

date

The required date argument is any Variant, numeric expression, string expression, or any combination, that can represent a date. If date contains Null, Null is returned.

2.1.2 Month Function

Month (date)

Returns a Variant (Integer) specifying a whole number between 1 and 12, inclusive, representing the month of the year.

Arguments:

date

The required date argument is any Variant, numeric expression, string expression, or any combination, that can represent a date. If date contains Null, Null is returned.

2.1.3 Year Function

Year (*date*)

Returns a Variant (Integer) containing a whole number representing the year.
The required date argument is any Variant, numeric expression, string expression, or any combination, that can represent a date. If date contains Null, Null is returned.

2.1.4 Weekday Function

Weekday (*date*, [*firstdayofweek*])

Returns a Variant (Integer) (or constant) containing a whole number representing the day of the week.

<i>Constant</i>	<i>Value</i>	<i>Description</i>
<i>vbSunday</i>	<i>1</i>	<i>Sunday</i>
<i>vbMonday</i>	<i>2</i>	<i>Monday</i>
<i>vbTuesday</i>	<i>3</i>	<i>Tuesday</i>
<i>vbWednesday</i>	<i>4</i>	<i>Wednesday</i>
<i>vbThursday</i>	<i>5</i>	<i>Thursday</i>
<i>vbFriday</i>	<i>6</i>	<i>Friday</i>
<i>vbSaturday</i>	<i>7</i>	<i>Saturday</i>

Arguments:

date

Required. Variant, numeric expression, string expression, or any combination, that can represent a date. If date contains Null, Null is returned.

firstdayofweek

Optional. A constant that specifies the first day of the week. If not specified, vbSunday is assumed.

The firstdayofweek argument has these settings:

<i>Constant</i>	<i>Value</i>	<i>Description</i>
<i>vbUseSystem</i>	<i>0</i>	<i>Use NLS (National Language Support) API setting.</i>
<i>vbSunday</i>	<i>1</i>	<i>Sunday (default)</i>
<i>vbMonday</i>	<i>2</i>	<i>Monday</i>
<i>vbTuesday</i>	<i>3</i>	<i>Tuesday</i>
<i>vbWednesday</i>	<i>4</i>	<i>Wednesday</i>
<i>vbThursday</i>	<i>5</i>	<i>Thursday</i>
<i>vbFriday</i>	<i>6</i>	<i>Friday</i>
<i>vbSaturday</i>	<i>7</i>	<i>Saturday</i>

2.1.5 WeekdayName Function

WeekdayName (*weekday*, [*abbreviate*] [, *firstdayofweek*])

Returns a string indicating the specified day of the week.

Arguments:

weekday

Required. The numeric designation for the day of the week. Numeric value of each day depends on setting of the firstdayofweek setting.

abbreviate

Optional. Boolean value that indicates if the weekday name is to be abbreviated. If omitted, the default is False, which means that the weekday name is not abbreviated.

firstdayofweek

Optional. Numeric value indicating the first day of the week.

<i>Constant</i>	<i>Value</i>	<i>Description</i>
<i>vbUseSystem</i>	<i>0</i>	<i>Use National Language Support (NLS) API setting.</i>
<i>vbSunday</i>	<i>1</i>	<i>Sunday (default)</i>
<i>vbMonday</i>	<i>2</i>	<i>Monday</i>
<i>vbTuesday</i>	<i>3</i>	<i>Tuesday</i>
<i>vbWednesday</i>	<i>4</i>	<i>Wednesday</i>
<i>vbThursday</i>	<i>5</i>	<i>Thursday</i>
<i>vbFriday</i>	<i>6</i>	<i>Friday</i>
<i>vbSaturday</i>	<i>7</i>	<i>Saturday</i>

2.1.6 MonthName Function

MonthName (*month*[, *abbreviate*])

Returns a string indicating the specified month.

Arguments:

Month

Required. The numeric designation of the month. For example, January is 1, February is 2, and so on.

Abbreviate

Optional. Boolean value that indicates if the month name is to be abbreviated. If omitted, the default is False, which means that the month name is not abbreviated.

2.2 Return a date

2.2.1 DateSerial Function

DateSerial (*year*, *month*, *day*)

Returns a Variant (Date) for a specified year, month, and day.

To specify a date, such as December 31, 1991, the range of numbers for each DateSerial argument should be in the accepted range for the unit; that is, 1–31 for days and 1–12 for months. However, you can also specify relative dates for each argument using any numeric expression that represents some number of days, months, or years before or after a certain date.

The following example uses numeric expressions instead of absolute date numbers. Here the DateSerial function returns a date that is the day before the first day (1 - 1), two months before August (8 - 2), 10 years before 1990 (1990 - 10); in other words, May 31, 1980.

DateSerial(1990 - 10, 8 - 2, 1 - 1)

For the year argument, values between 0 and 29, inclusive, are interpreted as the years 2000–2029. Values between 30 and 99 are interpreted as the years 1930–1999. For all other year arguments, use a four-digit year (for example, 1800).

When any argument exceeds the accepted range for that argument, it increments to the next larger unit as appropriate. For example, if you specify 35 days, it is evaluated as one month and some number of days, depending on where in the year it is applied. If any single argument is outside the range -32,768 to 32,767, an error occurs. If the date specified by the three arguments falls outside the acceptable range of dates, an error occurs.

Arguments:

year

Required; Integer. Number between 100 and 9999, inclusive, or a numeric expression.

month

Required; Integer. Any numeric expression.

day

Required; Integer. Any numeric expression.

2.2.2 DateValue Function

DateValue (date)

Returns a Variant (Date).

Arguments:

date

The required date argument is normally a string expression representing a date from January 1, 100 through December 31, 9999. However, date can also be any expression that can represent a date, a time, or both a date and time, in that range.

Remarks

If date is a string that includes **only numbers separated by valid date separators**, DateValue recognizes the order for month, day, and year according to the **Short Date** format you specified for your system. DateValue also recognizes unambiguous dates that contain month names, either in long or abbreviated form. For example, in addition to recognizing 12/30/1991 and 12/30/91, DateValue also recognizes December 30, 1991 and Dec 30, 1991.

If the year part of date is omitted, DateValue uses the current year from your computer's system date.

If the date argument includes time information, DateValue doesn't return it. However, if date includes invalid time information (such as "89:98"), an error occurs.

2.3 Set values

2.3.1 Date Statement

Date = date

Sets the current system date.

For systems running Microsoft Windows 95, the required date specification must be a date from January 1, 1980 through December 31, 2099. For systems running Microsoft Windows NT, date must be a date from January 1, 1980 through December 31, 2079.

3 Get values from Time

3.1 Extract values

3.1.1 Hour Function

Hour (time)

Returns a Variant (Integer) specifying a whole number between 0 and 23, inclusive, representing the hour of the day.

Arguments:

time

The required time argument is any Variant, numeric expression, string expression, or any combination, that can represent a time. If time contains Null, Null is returned.

3.1.2 Minute Function

Minute (time)

Returns a Variant (Integer) specifying a whole number between 0 and 59, inclusive, representing the minute of the hour.

Arguments:

time

The required time argument is any Variant, numeric expression, string expression, or

any combination, that can represent a time. If time contains Null, Null is returned.

3.1.3 Second Function

Second (*time*)

Returns a Variant (Integer) specifying a whole number between 0 and 59, inclusive, representing the second of the minute.

Arguments:

time

The required time argument is any Variant, numeric expression, string expression, or any combination, that can represent a time. If time contains Null, Null is returned.

3.2 Return a time

3.2.1 TimeSerial Function

TimeSerial (*hour, minute, second*)

Returns a Variant (Date) containing the time for a specific hour, minute, and second. To specify a time, such as 11:59:59, the range of numbers for each TimeSerial argument should be in the normal range for the unit; that is, 0–23 for hours and 0–59 for minutes and seconds. However, you can also specify relative times for each argument using any numeric expression that represents some number of hours, minutes, or seconds before or after a certain time. The following example uses expressions instead of absolute time numbers. The TimeSerial function returns a time for 15 minutes before (-15) six hours before noon (12 - 6), or 5:45:00 A.M.
TimeSerial(12 - 6, -15, 0)

When any argument exceeds the normal range for that argument, it increments to the next larger unit as appropriate. For example, if you specify 75 minutes, it is evaluated as one hour and 15 minutes. If any single argument is outside the range -32,768 to 32,767, an error occurs. If the time specified by the three arguments causes the date to fall outside the acceptable range of dates, an error occurs.

Arguments:

hour

Required; Variant (Integer). Number between 0 (12:00 A.M.) and 23 (11:00 P.M.), inclusive, or a numeric expression.

minute

Required; Variant (Integer). Any numeric expression.

second

Required; Variant (Integer). Any numeric expression.

3.2.2 TimeValue Function

TimeValue (*time*)

Returns a Variant (Date) containing the time.

Arguments:

Time

The required time argument is normally a string expression representing a time from 0:00:00 (12:00:00 A.M.) to 23:59:59 (11:59:59 P.M.), inclusive. However, time can also be any expression that represents a time in that range. If time contains Null, Null is returned.

You can enter valid times using a 12-hour or 24-hour clock. For example, "2:24PM" and "14:24" are both valid time arguments.

If the time argument contains date information, TimeValue doesn't return it. However, if

time includes invalid date information, an error occurs.

3.3 Set values

3.3.1 Time Statement

Time = *time*

Sets the system time.

Arguments:

Time

The required time argument is any numeric expression, string expression, or any combination, that can represent a time.

Remarks

If time is a string, Time attempts to convert it to a time using the time separators you specified for your system. If it can't be converted to a valid time, an error occurs.

4 Date and Time calculations

4.1 DateAdd Function

DateAdd (*interval*, *number*, *date*)

Returns a Variant (Date) containing a date to which a specified time interval has been added.

You can use the DateAdd function to add or subtract a specified time interval from a date. For example, you can use DateAdd to calculate a date 30 days from today or a time 45 minutes from now.

To add days to date, you can use Day of Year ("y"), Day ("d"), or Weekday ("w").

The DateAdd function won't return an invalid date. The following example adds one month to January 31:

DateAdd("m", 1, "31-Jan-95")

In this case, DateAdd returns 28-Feb-95, not 31-Feb-95. If date is 31-Jan-96, it returns 29-Feb-96 because 1996 is a leap year.

Note: The format of the return value for DateAdd is determined by Control Panel settings, not by the format that is passed in date argument.

Arguments:

interval

Required. String expression that is the interval of time you want to add.

<i>Setting</i>	<i>Description</i>
<i>yyyy</i>	<i>Year</i>
<i>q</i>	<i>Quarter</i>
<i>m</i>	<i>Month</i>
<i>y</i>	<i>Day of year</i>
<i>d</i>	<i>Day</i>
<i>w</i>	<i>Weekday</i>
<i>ww</i>	<i>Week</i>
<i>h</i>	<i>Hour</i>
<i>n</i>	<i>Minute</i>
<i>s</i>	<i>Second</i>

number

Required. Numeric expression that is the number of intervals you want to add. It can be positive (to get dates in the future) or negative (to get dates in the past).

If number isn't a Long value, it is rounded to the nearest whole number before being evaluated.

date

Required. Variant (Date) or literal representing date to which the interval is added.
If the calculated date would precede the year 100 (that is, you subtract more years than are in date), an error occurs.

4.2 DateDiff Function

DateDiff (*interval*, *date1*, *date2*[, *firstdayofweek*[, *firstweekofyear*]])

Returns a Variant (Long) specifying the number of time intervals between two specified dates.

You can use the DateDiff function to determine how many specified time intervals exist between two dates. For example, you might use DateDiff to calculate the number of days between two dates, or the number of weeks between today and the end of the year.

To calculate the number of days between date1 and date2, you can use either Day of year ("y") or Day ("d"). When interval is Weekday ("w"), DateDiff returns the number of weeks between the two dates. If date1 falls on a Monday, DateDiff counts the number of Mondays until date2. It counts date2 but not date1. If interval is Week ("ww"), however, the DateDiff function returns the number of calendar weeks between the two dates. It counts the number of Sundays between date1 and date2. DateDiff counts date2 if it falls on a Sunday; but it doesn't count date1, even if it does fall on a Sunday. When comparing December 31 to January 1 of the immediately succeeding year, DateDiff for Year ("yyyy") returns 1 even though only a day has elapsed.

Arguments:

interval

Required. String expression that is the interval of time you use to calculate the difference between date1 and date2.

<i>Setting</i>	<i>Description</i>
<i>yyyy</i>	<i>Year</i>
<i>q</i>	<i>Quarter</i>
<i>m</i>	<i>Month</i>
<i>y</i>	<i>Day of year</i>
<i>d</i>	<i>Day</i>
<i>w</i>	<i>Weekday</i>
<i>ww</i>	<i>Week</i>
<i>h</i>	<i>Hour</i>
<i>n</i>	<i>Minute</i>
<i>s</i>	<i>Second</i>

date1, *date2*

Required; Variant (Date). Two dates you want to use in the calculation.

If date1 refers to a later point in time than date2, the DateDiff function returns a negative number.

If date1 or date2 is a date literal, the specified year becomes a permanent part of that date. However, if date1 or date2 is enclosed in double quotation marks (" "), and you omit the year, the current year is inserted in your code each time the date1 or date2 expression is evaluated. This makes it possible to write code that can be used in different years.

firstdayofweek

Optional. A constant that specifies the first day of the week. If not specified, Sunday is assumed.

The firstdayofweek argument affects calculations that use the "w" and "ww" interval symbols.

<i>Constant</i>	<i>Value</i>	<i>Description</i>
<i>vbUseSystem</i>	<i>0</i>	<i>Use the NLS API setting.</i>
<i>vbSunday</i>	<i>1</i>	<i>Sunday (default)</i>
<i>vbMonday</i>	<i>2</i>	<i>Monday</i>
<i>vbTuesday</i>	<i>3</i>	<i>Tuesday</i>
<i>vbWednesday</i>	<i>4</i>	<i>Wednesday</i>
<i>vbThursday</i>	<i>5</i>	<i>Thursday</i>
<i>vbFriday</i>	<i>6</i>	<i>Friday</i>
<i>vbSaturday</i>	<i>7</i>	<i>Saturday</i>

firstweekofyear

Optional. A constant that specifies the first week of the year. If not specified, the first week is assumed to be the week in which January 1 occurs.

<i>Constant</i>	<i>Value</i>	<i>Description</i>
<i>vbUseSystem</i>	<i>0</i>	<i>Use the NLS API setting.</i>
<i>vbFirstJan1</i>	<i>1</i>	<i>Start with week in which January 1 occurs (default).</i>
<i>vbFirstFourDays</i>	<i>2</i>	<i>Start with the first week that has at least four days in the new year.</i>
<i>vbFirstFullWeek</i>	<i>3</i>	<i>Start with first full week of the year.</i>

4.3 DatePart Function

DatePart (*interval*, *date*[,*firstdayofweek*[, *firstweekofyear*]])

Returns a Variant (Integer) containing the specified part of a given date. You can use the DatePart function to evaluate a date and return a specific interval of time. For example, you might use DatePart to calculate the day of the week or the current hour.

Arguments:

interval

Required. String expression that is the interval of time you want to return.

<i>Setting</i>	<i>Description</i>
<i>yyyy</i>	<i>Year</i>
<i>q</i>	<i>Quarter</i>
<i>m</i>	<i>Month</i>
<i>y</i>	<i>Day of year</i>
<i>d</i>	<i>Day</i>
<i>w</i>	<i>Weekday</i>
<i>ww</i>	<i>Week</i>
<i>h</i>	<i>Hour</i>
<i>n</i>	<i>Minute</i>
<i>s</i>	<i>Second</i>

date

Required. Variant (Date) value that you want to evaluate.

If date is a date literal, the specified year becomes a permanent part of that date. However, if date is enclosed in double quotation marks (" "), and you omit the year, the current year is inserted in your code each time the date expression is evaluated. This makes it possible to write code that can be used in different years.

firstdayofweek

Optional. A constant that specifies the first day of the week. If not specified, Sunday is assumed.

The firstdayofweek argument affects calculations that use the "w" and "ww" interval symbols.

<i>Constant</i>	<i>Value</i>	<i>Description</i>
<i>vbUseSystem</i>	<i>0</i>	<i>Use the NLS API setting.</i>
<i>vbSunday</i>	<i>1</i>	<i>Sunday (default)</i>
<i>vbMonday</i>	<i>2</i>	<i>Monday</i>
<i>vbTuesday</i>	<i>3</i>	<i>Tuesday</i>
<i>vbWednesday</i>	<i>4</i>	<i>Wednesday</i>
<i>vbThursday</i>	<i>5</i>	<i>Thursday</i>
<i>vbFriday</i>	<i>6</i>	<i>Friday</i>
<i>vbSaturday</i>	<i>7</i>	<i>Saturday</i>

firstweekofyear

Optional. A constant that specifies the first week of the year. If not specified, the first week is assumed to be the week in which January 1 occurs.

<i>Constant</i>	<i>Value</i>	<i>Description</i>
<i>vbUseSystem</i>	<i>0</i>	<i>Use the NLS API setting.</i>
<i>vbFirstJan1</i>	<i>1</i>	<i>Start with week in which January 1 occurs (default).</i>
<i>vbFirstFourDays</i>	<i>2</i>	<i>Start with the first week that has at least four days in the new year.</i>
<i>vbFirstFullWeek</i>	<i>3</i>	<i>Start with first full week of the year.</i>

5 Formatting

5.1 Format Function

Format (*expression*[, *format*[, *firstdayofweek*[, *firstweekofyear*]]])

Returns a Variant (String) containing an expression formatted according to instructions contained in a format expression.

To Format Dates and times do this: Use predefined named date/time formats or create user-defined date/time formats.

To Format Date and time serial numbers do this: Use date and time formats or numeric formats.

Arguments:

expression

Required. Any valid expression.

format

Optional. A valid named or user-defined format expression.

User-Defined Date/Time Formats:

Date:

Character	Description
<i>(/)</i>	Date separator. In some locales, other characters may be used to represent the date separator. The date separator separates the day, month, and year when date values are formatted. The actual character used as the date separator in formatted output is determined by your system settings.
<i>d</i>	Display the day as a number without a leading zero (1 - 31).
<i>dd</i>	Display the day as a number with a leading zero (01 - 31).
<i>ddd</i>	Display the day as an abbreviation (Sun - Sat).
<i>dddd</i>	Display the day as a full name (Sunday - Saturday).
<i>dddddd</i>	Display the date as a complete date (including day, month, and year), formatted according to your system's short date format setting. The default short date format is m/d/yy.
<i>ww</i>	Display a date serial number as a complete date (including day, month, and year) formatted according to the long date setting recognized by your system. The default long date format is mmmm dd, yyyy.
<i>w</i>	Display the day of the week as a number (1 for Sunday through 7 for Saturday).
<i>ww</i>	Display the week of the year as a number (1 - 54).
<i>m</i>	Display the month as a number without a leading zero (1 - 12). If m immediately follows h or hh, the minute rather than the month is displayed.
<i>mm</i>	Display the month as a number with a leading zero (01 - 12). If m immediately follows h or hh, the minute rather than the month is displayed.
<i>mmm</i>	Display the month as an abbreviation (Jan - Dec).
<i>mmmm</i>	Display the month as a full month name (January - December).
<i>q</i>	Display the quarter of the year as a number (1 - 4).
<i>y</i>	Display the day of the year as a number (1 - 366).
<i>yy</i>	Display the year as a 2-digit number (00 - 99).
<i>yyyy</i>	Display the year as a 4-digit number (100 - 9999).

Time

Character	Description
<i>(:)</i>	Time separator. In some locales, other characters may be used to represent the time separator. The time separator separates hours, minutes, and seconds when time values are formatted. The actual character used as the time separator in formatted output is determined by your system settings.
<i>h</i>	Display the hour as a number without leading zeros (0 - 23).
<i>hh</i>	Display the hour as a number with leading zeros (00 - 23).
<i>n</i>	Display the minute as a number without leading zeros (0 - 59).
<i>nn</i>	Display the minute as a number with leading zeros (00 - 59).
<i>s</i>	Display the second as a number without leading zeros (0 - 59).
<i>ss</i>	Display the second as a number with leading zeros (00 - 59).
<i>ttttt</i>	Display a time as a complete time (including hour, minute, and second), formatted using the time separator defined by the time format recognized by your system. A leading zero is displayed if the leading zero option is selected and the time is before 10:00 A.M. or P.M. The default time format is h:mm:ss.
<i>AM/PM</i>	Use the 12-hour clock and display an uppercase AM with any hour before noon; display an uppercase PM with any hour between noon and 11:59 P.M.
<i>am/pm</i>	Use the 12-hour clock and display a lowercase AM with any hour before noon; display a lowercase PM with any hour between noon and 11:59 P.M.

A/P	Use the 12-hour clock and display an uppercase A with any hour before noon; display an uppercase P with any hour between noon and 11:59 P.M.
a/p	Use the 12-hour clock and display a lowercase A with any hour before noon; display a lowercase P with any hour between noon and 11:59 P.M.
AMPM	Use the 12-hour clock and display the AM string literal as defined by your system with any hour before noon; display the PM string literal as defined by your system with any hour between noon and 11:59 P.M. AMPM can be either uppercase or lowercase, but the case of the string displayed matches the string as defined by your system settings. The default format is AM/PM.

Date and Time

Character	Description
c	Display the date as ddddd and display the time as ttttt, in that order. Display only date information if there is no fractional part to the date serial number; display only time information if there is no integer portion.

The following table identifies the predefined date and time format names:

Format Name	Description
General Date	Display a date and/or time. For real numbers, display a date and time, for example, 4/3/93 05:34 PM. If there is no fractional part, display only a date, for example, 4/3/93. If there is no integer part, display time only, for example, 05:34 PM. Date display is determined by your system settings.
Long Date	Display a date according to your system's long date format.
Medium Date	Display a date using the medium date format appropriate for the language version of the host application.
Short Date	Display a date using your system's short date format.
Long Time	Display a time using your system's long time format; includes hours, minutes, seconds.
Medium Time	Display time in 12-hour format using hours and minutes and the AM/PM designator.
Short Time	Display a time using the 24-hour format, for example, 17:45.

firstdayofweek

Optional. A constant that specifies the first day of the week.

Constant	Value	Description
vbUseSystem	0	Use the NLS API setting.
vbSunday	1	Sunday (default)
vbMonday	2	Monday
vbTuesday	3	Tuesday
vbWednesday	4	Wednesday
vbThursday	5	Thursday
vbFriday	6	Friday
vbSaturday	7	Saturday

firstweekofyear

Optional. A constant that specifies the first week of the year.

<i>Constant</i>	<i>Value</i>	<i>Description</i>
<i>vbUseSystem</i>	<i>0</i>	<i>Use the NLS API setting.</i>
<i>vbFirstJan1</i>	<i>1</i>	<i>Start with week in which January 1 occurs (default).</i>
<i>vbFirstFourDays</i>	<i>2</i>	<i>Start with the first week that has at least four days in the new year.</i>
<i>vbFirstFullWeek</i>	<i>3</i>	<i>Start with first full week of the year.</i>

5.2 FormatDateTime Function

FormatDateTime (*Date*[,*NamedFormat*])

Returns an expression formatted as a date or time.

Arguments:

Date

Required. Date expression to be formatted.

NamedFormat

Optional. Numeric value that indicates the date/time format used. If omitted, vbGeneralDate is used.

The NamedFormat argument has the following settings:

<i>Constant</i>	<i>Value</i>	<i>Description</i>
<i>vbGeneralDate</i>	<i>0</i>	<i>Display a date and/or time. If there is a date part, display it as a short date. If there is a time part, display it as a long time. If present, both parts are displayed.</i>
<i>vbLongDate</i>	<i>1</i>	<i>Display a date using the long date format specified in your computer's regional settings.</i>
<i>vbShortDate</i>	<i>2</i>	<i>Display a date using the short date format specified in your computer's regional settings.</i>
<i>vbLongTime</i>	<i>3</i>	<i>Display a time using the time format specified in your computer's regional settings.</i>
<i>vbShortTime</i>	<i>4</i>	<i>Display a time using the 24-hour format (hh:mm).</i>